1. project plan (schedule)

See attached project plan

2. List of Team Name, Members, roles and responsibilites

DMSACWSWG

DMV, **M**icrosoft, **S**usquehanna **A**ddress **C**hange **W**eb **S**ervice **W**ork **G**roup

Team Members:

Name	Organization	E-Mail	Phone
Chuck O'Keeffe	Microsoft Corporation	chuckok@microsoft.com	(540) 247-8063
Hank Farlow	Microsoft Corporation	hankf@microsoft.com	(804) 307-9591
Beth DeHaven	Microsoft Corporation	bdehaven@microsoft.com	(804) 337-3262
William Strydom	DMV	dmvw1s@dmv.state.va.us	(804) 367-8402
Lana Shelley	DMV	dmvlss@dmv.state.va.us	(804) 367-2635
Don Kendrick	DMV	dmvdsk@dmv.state.va.us	(804) 367-8336
Scott Fowler	VIPNET	sfowler@fipnet.org	(804) 786-6220
Tim Mauney	DMV	Dmvt1m@dmv.state.va.us	(804) 367-1379
David Froggatt	DMV	Dmvdcf@dmv.state.va.us	(804) 367-2380

New members as of 6/1/2002:

Bill Vencil	**Susquehanna**	**Billv@susqtech.com**	**(540) 665 3427**
Glenn Hickman	**Susquehanna**	**Glennh@susqtech.com**	**(540) 723 8700**

Left DMV effective 6/14/2002:

Debbie Dodson	DMV	dmvdhd@dmv.state.va.us	(804) 367-9227

Roles and responsibilities

Product Manager

- Holder of the project requirements
- Coordinates activities between development team and work-group
- Drives project vision
- Manages expectations
- Drives feature identification and prioritization
- Develops, maintains, and executes the communications plan

Program Manager

- Manages overall process
- Manages resource allocation
- Manages project schedule and status reporting
- Manages project scope and specification
- Facilitates team communication and negotiation
- Drives overall critical trade-off decisions

Developer

- Builds and tests features to meet the specifications and expectations
- Participates in design

- Estimates time and effort to complete each feature
- Servers the team as a technology consultant

Tester

- Develops testing strategy, plans, and scripts
- Manages the build process
- Conducts tests to accurately determine the status of product development
- Participates in setting the quality bar

Logistics Management

- Acts as team advocate to operations
- Acts as operations advocate to the team
- Plans and manages product deployment
- Participates in design, focusing on manageability, supportability, and deployability
- Supports the product during testing

Analyst

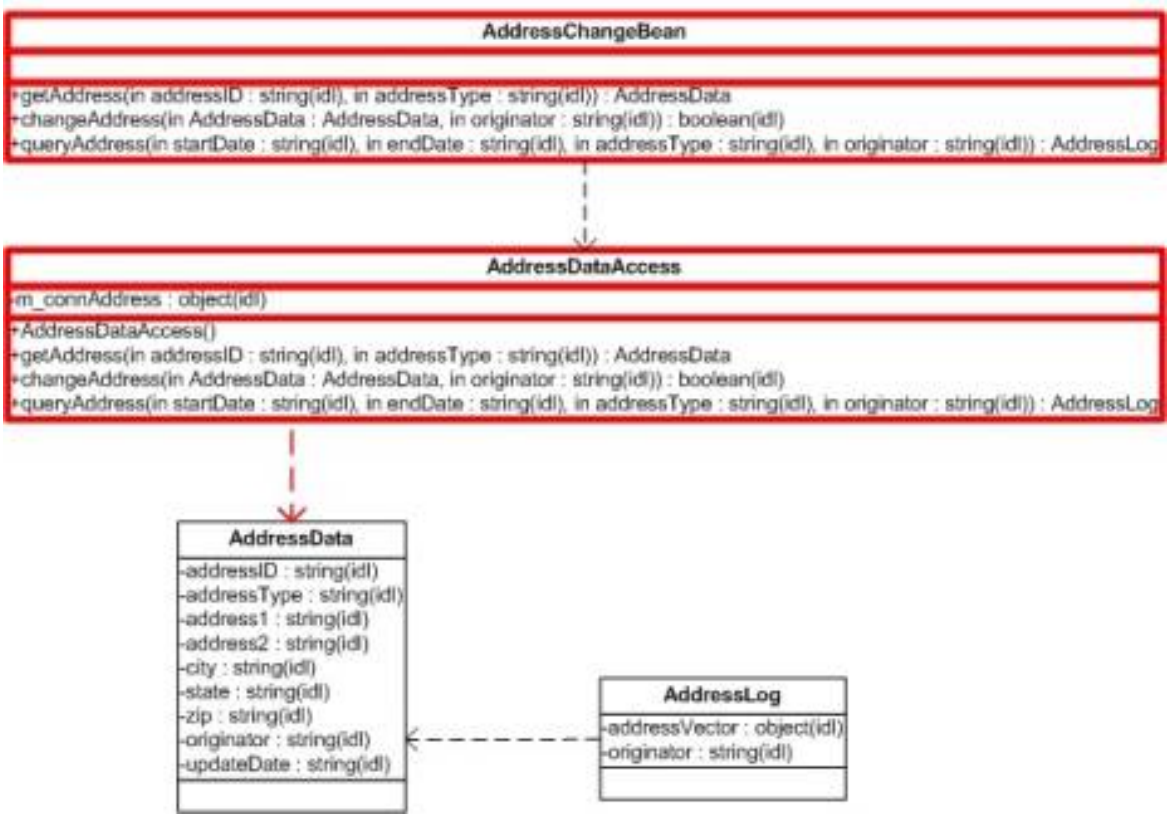
- Documents technical architecture of the development process
- Identifies and documents technical requirements outside of the scope of current development process
- Completes final report

3. proof-of-Concept design

Web Service Component: The overall design of the team's Web Services application was dictated by the workgroup WSDL. The system architecture is straightforward, utilizing the .NET Framework to

house the web services component. Here is a simple diagram of the overall architecture:

Following the specifications of the workgroups agreed upon WSDL, a component was created. The simple UML model on the next page provides the specific detail of the component:

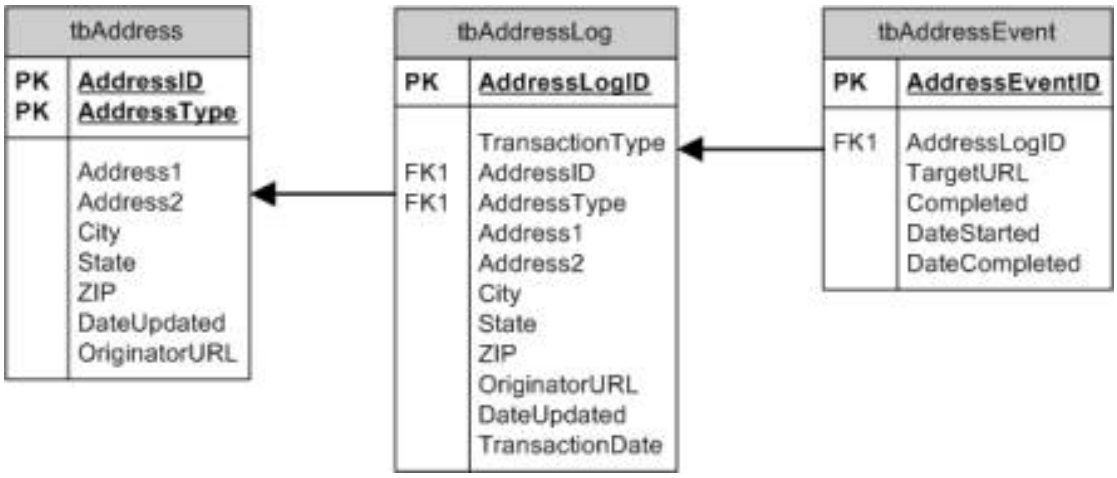


Database: A database schema was designed and implemented in the SQL Server 2000 environment. There are currently three tables in the schema, pending new requirements that may arise:

tbAddress: Repository for base address information

tbAddressLog: Maintains address change transaction for verification

tcAddressEvent: Address change events that must be processed by background process.



Background process: In order to automate the process of updating and querying the other teams' exposed web services, a scheduled background process is currently being designed. Its purpose will be to process all queued address changes events by delivering them to each address change Web Service found on the project UDDI server.

Client Test Application: In order to more easily test the functionality of the web service and manually create address changes, a desktop application is being developed using Visual Studio .NET.

4. Business/Functional requirements

§ Provide an address change function to the citizen, which will use web service in the background to update subscribers of the service of the address change information.

§ Address change information will be provided (pushed) to all subscribers in **a change address** function, if the change originated at the DMV.

§ A **get address** function will be provided, if a subscriber chooses to retrieve (pull) the address information.

§ A **list address changes** function will also be provide, if a subscriber chooses to retrieve (pull) all address changes done between a specified date time stamps for a specified agency/locality (originator).

§ Only address changes that originated at the DMV will be propagated to all subscribers. This will prevent loop of address change originating at other agencies/localities.

5. project equipment description

Hardware

§ 2 Compaq Proliant Servers ML Series

§ (2) 72 GB Hard Drives

§ 512 MB Memory

§ Internal tape

§ 2 – 17” monitors

§ 1 Workstation for application development and testing of remote access to web services

Compaq provided the hardware.

Software

§ Visual Studio.NET – Web Services Development

- All components for XML, WSDL, SOAP, UDDI

§ Windows 2000 Advanced Server for hosting platform. Also included in the installation is the .NET Framework to provide Web Services capabilities

§ SQL Server 2000 for database storage and analytic processing of data

Microsoft provided the software and hardware.

Location

§ The servers are housed at VIPNET.

6. acquisition and installation activities

Equipment was acquired by Microsoft for the purposes of the pilot project.

7. server requirements

There are no additional requirements above and beyond the equipment described above.

8. networking requirements

There are no specific networking requirements other than the following:

§ Both Compaq servers must be on the same network segment.

§ The web server must be available on the Internet allowing connections via the HTTP protocol and ports 80 and 443 (inbound and outbound).

9. training requirements

§ Tim Mauney and William Strydom from the DMV were sent to the .NET training provided by Microsoft at DIT on June 6 & 7.

§ No other training is required for this project.

10. testing Plan

Currently, Susquehanna Technologies is following its standard method of component testing. In addition to our internal testing, we suggest the following component tests and scenarios during the interoperability phase of the project:

1. getAddress()

a. Each team will successfully query all address change Web Services for a specified list of addresses and log return results.

b. Each team will produce a report of all addresses changed during the test.

2. changeAddress()

a. Each team will successfully utilize the changeAddress() method to change a specified list of addresses (unique to the team) in its internal system and log results.

b. Each team will successfully utilize the changeAddress() method on all remote address change Web Services for the list of changed addresses in step A. Each remote system will log the delivered address change.

c. Each team will produce a report of all addresses changes, both internally and externally.

3. queryAddress()

a. Each team will retrieve a list of addresses changed for a specified period of time from all remote systems.

b. Each team will provide a report of addresses retrieved, which will be verified by remote system teams.

11. Development / technical

No outstanding issues at this time.

12. Protocol / specification ISSUES

When using Microsoft's WSDL utility, some minor errors were encountered when converting the workgroup WSDL that was provided by BEA. Some minor editing of the utility's output corrected the issue.

13. Interoperability issues

None encountered at the present time. The .NET-based Web Service has been successfully tested from a PERL environment using Soap Lite. Here is the sample code used and resulting output:

Callfromperl.pl Source Code:

```
use SOAP::Lite;

my $param1 = SOAP::Data->type(string => "2112")->name('string0');;

my $param2 = SOAP::Data->type(string => "individual")->name('string1');

my $objResult = SOAP::Lite

  -> uri('http://localhost/DMVAddressWebService')

  -> on_action(sub{sprintf '%s/%s', @_ })

  -> proxy('http://localhost/DMVAddressWebService/AddressChangeBean.asmx')

  -> getAddress( $param1, $param2 )

  -> result or die "Failed to attach to web service";

print "AddressID: $objResult->{addressID}\n";

print "AddressType: $objResult->{addressType}\n";

print "Address1: $objResult->{address1}\n";

print "Address2: $objResult->{address2}\n";
```

```
print "City: $objResult->{city}\n";

print "State: $objResult->{state}\n";

print "ZIP: $objResult->{zip}\n";

print "Originator: $objResult->{originator}\n";

print "Update Date: $objResult->{updateDate}\n";
```

Output:

```
C:\DATA\Projects\MCS\DMV>perl callfromperl.pl

AddressID: 2112

AddressType: individual

Address1: 3668 Apple Pie Ridge Road

Address2:

City: Winchester

State: VA

ZIP: 22601

Originator: http://localhost/DMVAddressWebService

Update Date: 2002-05-08T00:00:00.0000000-04:00
```

14. “Best Practice” comments

§ When converting a WSDL created from another development platform, manually verify that the WSDL generation was correct.

§ Keep the solution as simple as possible while still meeting the design requirements. This will help speed up the process of development and guarantee success.

§ Be aware that a Web Service is a loosely coupled, disconnected process. Session state is an issue that needs to be considered carefully. While a concern, this is very similar in concept to the general practice of web application development.

§ General Microsoft platform “best practices” should be followed in any type of Microsoft-based development efforts.

§ Naming of parameters in the WSDL should not use common terms that may be reserved words. For example “string” is commonly used as a datatype.

15. Other Concerns/Issues

§ We would like to suggest that architects/developers from each team be encouraged to communicate details of implementation and deployment before interoperability testing begins. In addition, all teams should agree upon a common method of communication that keeps everyone “in the loop”.

§ Specific UDDI configuration information is necessary to complete the background process application.

16. Other Comments

None at this time.

17. Cost/Time Estimate

1) Meetings:

05/15/2002	DMV & Microsoft work group	1½ hours at DMV
05/22/2002	DMV & Microsoft work group	1½ hours at DMV
05/29/2002	DMV & Microsoft work group	1½ hours at DMV
05/30/2002	All work groups	8 hours at VRS
06/04/2002	DMV, Microsoft & Susquehanna work group	1 hour at DMV
06/10/2002	DMV & Susquehanna work group	2 hours at DMV
06/20/2002	All work groups	4 hours at VRS

2) Development:

6/20/2002	Susquehanna development completed	35 hours at SusQtech

3) Training:

06/06/2002	.NET training	8 hours at DIT
06/07/2002	.NET training	8 hours at DIT

4) Testing: